Een ontwerp dat de organisatie kan bijbenen

de Boer, T.W.; van Uitert, Johannes

Published in:
Informatie

Een ontwerp dat de organisatie kan bijbenen

Vetorecht bij de deskundige

Informatie, maandblad voor de informatievoorziening, ISSN 00199907, januari/februari 2004, jaargang 46, 01-02-2004

Een pas opgeleverd informatiesysteem sluit vaak niet aan bij het werk van de gebruiker. Bovendien is het star, doordat de processen die het ondersteunt onwrikbaar zijn vastgelegd. De door de auteurs gepresenteerde methode moet aan deze bezwaren tegemoetkomen. Volgens deze methode stuurt de gebruiker het ontwikkelproject.

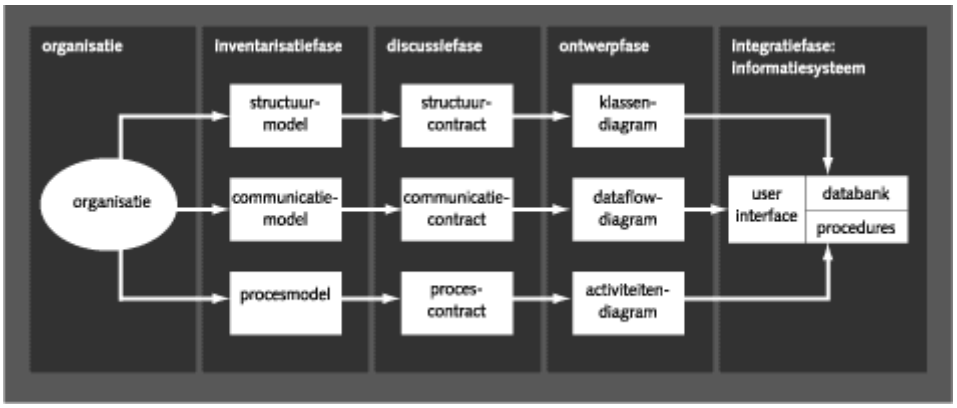
De methode haakt aan bij principes die voorkomen in Extreme Programming, Agile Modeling en Use-case Driven Modeling. Uitgangspunt is dat gedurende de gehele levenscyclus van het informatiesysteem de domeindeskundige de sturende partij is. Dus niet alleen in de eerste fasen van het project, maar ook bij het bouwen van het systeem en het onderhoud daarna. Dat betekent niet dat de domeindeskundige de ict-specialist vervangt. Dat kan niet. De ict-specialist heeft zijn eigen deskundigheid en die kan niet worden gemist. Wel kan de domeindeskundige een gedetailleerde beschrijving geven van de organisatie en specifieke processen daarbinnen, zodat de ict-specialist met zijn systeemmodellen daarop aansluit. Dit waarborgt dat wat de domeindeskundige wil, daadwerkelijk in het informatiesysteem wordt gerealiseerd. De methode ondersteunt de domeindeskundige bij het modelleren van de organisatie en het specificeren van de functionaliteit. Dit vormt de basis voor het latere functionele en logische ontwerp van de ict-specialist.

Drie modellen en vier fasen

Modelleren van organisaties kan op vele verschillende manieren (Morgan, 1997). Uitgangspunt voor de methode is de definitie van Vernadat (1996), die een enterprise model omschrijft als 'a consistent set of special purpose and complementary models, describing various facets of an enterprise to satisfy some purpose of some business users'. In de methode beperken we de modellen tot drie soorten, die goed aansluiten bij gebruikelijke modellering in ict en bij de belevingswereld van de domeindeskundige. De drie soorten zijn het statische structuurmodel, het communicatiemodel en het procesmodel. Samen vormen ze het domeinmodel. Het statische structuurmodel geeft een inventarisatie van de relevante objecten en klassen in het domein en de wijze waarop ze aan elkaar zijn gerelateerd. Het communicatiemodel beschrijft de communicatielijnen in de organisatie en de berichten die worden verzonden. Het procesmodel beschrijft de processen die zich in de organisatie afspelen.

De ontwikkeling van ieder van deze drie modellen doorloopt vier fasen. De eerste fase is een inventarisatiefase waarin alle relevante onderdelen van de modellen worden beschreven. De tweede fase is een afronding van de eerste: de modellen komen definitief vast te staan en krijgen nu het karakter van een contractmodel of modelcontract. Daarbij wordt vooral afgesproken welke functionaliteit het informatiesysteem krijgt en wie de gebruikers zijn. In

de derde fase worden de contractmodellen vertaald naar gangbare modellen voor systeemontwerp. Samen vormen ze dan het ontwerpmodel. In de vierde fase worden de drie modellen geïntegreerd in het ontwerp voor het informatiesysteem. Figuur 1 geeft een overzicht van de modellen in de verschillende fasen.



Figuur 1 - Drie modellen in vier fasen

Een aandachtspunt is dat de modellen in de verschillende fasen naadloos op elkaar aansluiten. De verantwoordelijkheid voor de besluitvorming komt bij de domeinspecialist te liggen. Daartoe bevat de methode een formele discussiefase, waarin deelnemers voorstellen tot verandering kunnen inbrengen. In de discussie vallen besluiten over de inhoud van de modellen. Dit is de fase waarin de contractmodellen worden vastgesteld.

De discussiefase is de trait d'union tussen de modellen van de domeindeskundige en de ontwerpmodellen. De domeindeskundige heeft hierin het laatste woord. Hij bepaalt uiteindelijk of een model wordt aangepast of niet. Hier maken we de ict-specialist ondergeschikt aan de domeindeskundige. Wanneer een ict-specialist een verandering in een ontwerpmodel wil aanbrengen, mag dat als het domeinmodel daardoor niet verandert. Dat laatste gebeurt alleen wanneer er bijvoorbeeld bij het opstellen van ontwerpdiagrammen omissies of fouten naar boven komen. In deze gevallen wordt het domeinmodel aangepast, mits de domeindeskundige akkoord gaat.

Een voorbeeld is een relatie tussen twee objecten in het structuurmodel dat in het resulterende klassendiagram een associatieklasse wordt. Klaarblijkelijk wil de domeindeskundige gegevens van de associatie bijhouden en dus is het terecht dat het domeinmodel wordt aangepast. Aanpassing is ook mogelijk in de vorm van uitbreiding. In dat geval wordt het domeinmodel niet noodzakelijkerwijs aangepast, maar kan het ontwerpmodel een nadere detaillering of specificatie geven. Voorbeelden hiervan zijn helper classes en boundary classes. Losjes geformuleerd mag de ict-specialist de beschrijving uit het domeinmodel dus wel uitbreiden maar niet wijzigen.

Modellen in dezelfde fase sluiten op elkaar aan door synchronisatie. Daarmee wordt de afstemming tussen de modellen gecontroleerd. Zo moeten de gegevens die in de berichten in de dataflowdiagrammen voorkomen, ook voorkomen in de attributen in het structuurdiagram. Een ander voorbeeld zijn de berichten die in de activiteitendiagrammen worden genoemd. Die moeten ook te vinden zijn in de dataflowdiagrammen. In de

inventarisatiefase zal deze synchronisatie losjes gebeuren, in de contractfase worden de drie contractmodellen definitief gesynchroniseerd.

De aansluiting tussen de fasen en de synchronisatie leiden ertoe dat bijvoorbeeld het klassendiagram een exacte afspiegeling is van het statische structuurmodel en dat het structuurcontract de data specificeert van de andere twee contractmodellen. Wanneer hier verschillen optreden, wijkt het resulterende informatiesysteem af van wat de domeindeskundige beoogt. Een van de manieren om naadloze aansluiting te bewerkstelligen is het gebruik van schematechnieken in de inventarisatiefase en een frequent gebruik van terugkoppeling. Omdat de domeindeskundige de schematechnieken vaak niet goed beheerst, is hier samenwerking nodig tussen domeindeskundige en informatieanalist.

Ook kan het dienstig zijn hier het credo van Agile Modeling (Ambler) te hanteren: een schematechniek is een middel waarbij het mogelijk is zo nodig af te wijken van formele voorschriften. Voorop blijft staan dat de domeindeskundige bepaalt hoe de modellen er uit gaan zien. De informatieanalist biedt ondersteuning bij het uitwerken van de modellen. In de praktijk blijkt het effectief te zijn om bijvoorbeeld een user story (Jeffries, 2000) uit te werken tot use case en activiteitendiagram en het laatste weer terug te koppelen naar de domeindeskundige.

De ratio achter de keuze voor deze drie soorten modellen ligt in het resultaat. De drie soorten leveren ieder een generiek onderdeel van het informatiesysteem. We onderscheiden bij een informatiesysteem: de databank, het user interface en de procedures. Het structuurmodel leidt tot een klassendiagram dat de basis legt voor de datastructuur. Het communicatiemodel geeft de datastromen tussen personen en functies in de organisatie. Deze worden in de derde fase beschreven in dataflowdiagrammen. Wanneer daarin de grenzen van het systeem komen vast te staan, is direct duidelijk wie met het systeem gaat communiceren en om welke data het hierbij gaat. Dat vormt de grondslag voor het user interface. De procesmodellen ten slotte leiden tot een reeks activiteitendiagrammen die samen procedures vormen. Dat rondt het functioneel ontwerp van het systeem af.

Hoewel ieder van de drie modellen leidt tot een formele representatie in de vorm van een diagram of andere formele beschrijving, is in de inventarisatiefase het gebruik van schematechnieken bewust vrij gehouden. De domeindeskundige moet zijn ideeën kwijt kunnen. Schematechnieken moeten daarbij helpen en niet in de weg staan. Dus laten we in de inventarisatiefase overtreding van de regels voor het hanteren van de techniek toe. Later trekken we dat recht, in de discussiefase en eventueel in de ontwerpfase. Daarbij blijft de aansluiting bij de modellen uit de inventarisatiefase bestaan. Wijzigingen dienen alleen om de modellen in de contract- en ontwerpfase formeel correct te maken. Het contract moet ondubbelzinnig zijn en voor iedere participant duidelijk zijn. Om allerlei redenen is het verstandig de drie modellen parallel te ontwikkelen. Eventueel kan men beginnen met de procesmodellen. Deze spreken domeindeskundigen in het algemeen het meest aan.

Het structuurmodel

Het structuurmodel geeft een statische beschrijving van het domein. Als schematechniek gebruiken we een uitgeklede vorm van het klassendiagram uit de Unified Modelling Language (UML) (Dennis, 2002; Larman, 2002), maar een entity-relationshipdiagram (Chen,

1976) zal even goed werken. Het belangrijkste is dat de domeindeskundige een beschrijving opstelt van de relevante objecten (of klassen van objecten, eventueel entiteiten), hun eigenschappen en hun relaties met andere (klassen van) objecten.

Hier is al direct duidelijk dat we in de inventarisatiefase vooral niet roomser willen zijn dan de paus. In de eerste inventariserende diagrammen laten we het gebruik van objecten naast klassen toe. Later worden deze uitgewerkt en ontstaan in de derde fase ordentelijke klassendiagrammen. De ervaring leert dat domeindeskundigen in eerste instantie moeite hebben met denken in klassen, objecten en attributen. Als introductie gebruiken we soms het organigram. De bladeren daarvan geven vaak een eerste indicatie van de objecten die van belang zijn. Daarnaast is het open interview erg belangrijk. Relevante objecten, klassen en relaties brengen we aan het licht door een grammaticale analyse (Kristen, 1998). In tegenstelling tot wat bij UML gebruikelijk is, voorzien we de klassen niet van operaties. Wanneer de geïnterviewden die spontaan noemen, worden ze wel geregistreerd. Echter, het is bij het structuurmodel het doel de datastructuur te specificeren. Operaties, acties en functies komen bij de andere modellen aan de orde.

Het structuurmodel leidt via het klassendiagram uiteindelijk tot een specificatie van de databasestructuur. In ons geval is dat altijd een relationele structuur, maar ieder andere basisstructuur zijn natuurlijk ook af te leiden. De specificaties in het structuurmodel en klassendiagram leiden tot een volledige data dictionary. Het goede daarvan is dat zij geformuleerd is in termen die de domeindeskundige heeft bedacht.

Het communicatiemodel

Het tweede beeld waarin de organisatie wordt gemodelleerd is het communicatiemodel. Vanaf de eerste inventariserende modellen gebruiken we hiervoor de techniek van dataflowdiagrammen (dfd). Een dfd spreekt gemakkelijk aan en domeindeskundigen hebben een goed idee van de data die worden overdragen. De gevolgde lijn is tamelijk klassiek: we beginnen met het contextdiagram dat een overzicht geeft van de gehele organisatie en de globale communicatie daarbinnen. Dit diagram wordt verder gespecificeerd tot alle relevante communicatie is geregistreerd. Daarbij moet men niet alleen denken aan ruimtelijke communicatie, maar ook aan communicatie in de tijd, zoals het gebruik van archieven. In de contractfase wordt duidelijk welke functies het informatiesysteem krijgt. Door analyse van de functies (personen) die daarmee communiceren, wordt duidelijk wie de gebruikers van het systeem zijn en wat zij van het systeem verwachten in termen van data en boodschappen. Op basis daarvan is het mogelijk het user interface op te zetten. Iedere gebruiksfunctie wordt een optie in het hoofdmenu, dat op klassieke wijze verder wordt gesplitst in submenu's. Op dit punt is invloed van de toekomstige gebruikers gewenst. De indeling in menu's en submenu's moet aansluiten bij hun denk- en werkwereld.

Het procesmodel

Processen omvatten activiteiten die mensen in bepaalde functies uitvoeren. Activiteiten zijn op vele manieren te ordenen. Wij kiezen voor een ordening waarbij activiteiten worden samengebracht als ze een gemeenschappelijk doel hebben (Eriksson & Penker, 2000). Dat levert vaak een te ruime ordening op. Bijvoorbeeld de doelstelling 'de klant tevreden

houden' geeft een veel te complex geheel van activiteiten. De doelstellingen komen daarom in een hiërarchie te staan, een boomstructuur. De bladeren van deze hiërarchische boomstructuur zijn veelal op voldoende detailniveau om goed afgebakende en overzichtelijke processen te vormen. Degenen die ze uitvoeren beschrijven de zo gevonden processen. Voor de beschrijving hebben we gekozen voor het concept van de user story's, die de domeindeskundige uitwerkt tot de use case uit UML. De use case geeft de auteur de kans om zijn verhaal in zijn eigen woorden te vertellen en geeft tegelijk voldoende structuur om de processen formeel op de kaart te zetten. Als use cases toch te complex worden, splitst de auteur ze alsnog in twee of meer aparte cases.

De ict-specialist werkt de use case daarna uit in een activiteitendiagram, met zorg voor de aansluiting tussen het verhaal en het diagram. Hij koppelt het diagram terug naar de auteur van de use case. Dat geeft discussie, het lost losse eindjes op en zo ontstaat een duidelijk beeld van wat een proces precies inhoudt, wie de actor is, wie het proces start et cetera. Als de discussie ten einde is, ligt er een duidelijke reeks procesmodellen op tafel, netjes uitgesplitst naar deelactiviteiten en weergegeven in activiteitendiagrammen. Deze vormen de basis voor de procedures in het informatiesysteem.

Integratie

In de vierde fase, de integratiefase, worden de drie soorten modellen samengevoegd tot één ontwerpmodel voor het informatiesysteem. Door het parallel ontwikkelen van drie soorten modellen, is de kans groot dat er discrepanties ontstaan, ondanks de beoogde aansluiting en synchronisatie. Daarom wordt in de integratiefase een laatste controle uitgevoerd. Daarbij wordt ieder model vergeleken met de andere en gecontroleerd op volledigheid. Zo moet bij iedere optie uit het user interface een procedure aanwezig zijn. Bij iedere procedure wordt gecontroleerd of de benodigde gegevens aanwezig zijn in de databank en of ze voldoende eenduidig zijn gedefinieerd. Zo waarborgt de integratiefase dat het systeem intern consistent is en het gehele gewenste domein afdekt.

Ontwikkeling in drie soorten modellen legt de basis voor een grote mate van onafhankelijkheid van de drie systeemonderdelen. Het user interface biedt allerhande opties: in principe start elke optie één procedure, eventueel omvat die subprocedures. De koppeling met de databank wordt even strikt geformuleerd: wanneer een scherm gegevens opvraagt, gebeurt dat in principe met één query. De datastructuur bevat de gegevens waarmee de procedures werken. Gebruikers starten procedures met menuopties. Het menu is hiërarchisch opgebouwd, vanuit het hoofdmenu zijn submenu's bereikbaar die eventueel verder zijn gedetailleerd in subsubmenu's. Uiteindelijk bevat een menu opties, die ieder een aparte procedure starten. De relatie tussen een menuoptie en een procedure wordt gelegd met een schermformulier. Idealiter start een menuoptie één procedure die werkt met één query op de database die niet meer dan één schermformulier beslaat. Dit is te realiseren door procedures op te delen in subprocedures die elkaar aanroepen.

Op deze wijze blijven de onderdelen van het informatiesysteem redelijk autonoom. De integratie is de afsluiting van het functioneel en logisch ontwerp. Deze ontwerpen vormen de basis voor het informatiesysteem.

Een voorbeeld

Hierboven is al gewezen op de relatieve zelfstandigheid van de systeemonderdelen. Daardoor is het mogelijk wijzigingen in het domein eenduidig aan te geven en door te voeren. Een voorbeeld. Wanneer een supermarktketen een bonuskaart voor klanten invoert, verandert er veel in de registraties en bij het afrekenen aan de kassa. Binnen onze methode is een en ander duidelijk te scheiden en zijn de noodzakelijke veranderingen eenduidig aan te geven.1 Dat gaat als volgt. Vanuit het structuurmodel blijkt dat de klasse klant verandert: er is nu sprake van een extra attribuut, het bonuskaartnummer. Producten die in de aanbieding zijn, moeten nu ook een bonusprijs krijgen. Deze veranderingen in het structuurmodel leiden tot aanpassing van de databank. In het procesmodel komt bij het afrekenen een opmerking over het verwerken van de bonuskaart. Als de klant een bonuskaart heeft, scant de kassier die en geldt voor de in aanmerking komende producten de bonusprijs of de bonuskorting.

Uit het communicatiemodel blijkt de verandering omdat daar een dataflow gaat van klant naar kassier: het bonuskaartnummer wordt als bericht verzonden. Dat betekent dat in het user interface bij het afrekenen ruimte moet zijn voor het invoeren van het kaartnummer. Een op zich grote verandering wordt op deze wijze overzichtelijk in kaart gebracht. Natuurlijk zijn er meer wijzigingen denkbaar bij de introductie van een bonuskaart. Misschien wil men omzetten per bonuskaarthouder bijhouden. Maar ook dan geldt, dat de scheiding in drie soorten modellen de overzichtelijkheid sterk bevordert en daardoor aanpassing goed mogelijk maakt en soepel doet verlopen.

Conclusies

Door de besturing van het project bij de domeindeskundige te leggen, garandeert de methode dat het resulterende informatiesysteem goed aansluit bij wat de domeindeskundigen voor ogen staat. Een wat zijdelings voordeel is, dat de domeindeskundigen intensief nadenken over de eigen organisatie en de manier van werken. Vaak leidt dat in de ontwerpfase al tot verbeteringen. Het feit dat de organisatie duidelijk in kaart is gebracht, heeft verder tot gevolg dat wijzigingen goed kunnen worden aangeduid. Omdat het systeem via de drie soorten modellen goed aansluit, zijn gewenste veranderingen nauwkeurig aan te geven. Dat opent de mogelijkheid dat het informatiesysteem meegroeit met veranderingen in de organisatie.

Een ander voordeel van de methode is, dat de mensen in de organisatie het gevoel krijgen dat het hun systeem is. Ze krijgen het niet van bovenaf opgelegd, niet van buitenaf binnengedragen, ze bedenken en ontwerpen het van binnenuit. De domeindeskundigen dragen verantwoordelijkheid en nemen hun taak serieus. Daardoor is de weerstand tegen het systeem, en die is er altijd wel bij een nieuw systeem, minimaal.

Een laatste voordeel is dat we, door gebruik te maken van drie modellen die parallel worden gespecificeerd, de oude strijdvraag tussen dataoriëntatie en procesoriëntatie bij systeemontwerp ontwijken.

Natuurlijk zijn er ook nadelen te noemen. De methode vergt in eerste instantie veel tijd van domeindeskundigen. In de regel zijn dat de werknemers in de organisatie. Die moeten tijd vrijmaken voor het project en dat gaat niet zonder moeite. Hoewel de initiële investering niet gering is, denken we dat op wat langere termijn de voordelen de nadelen meer dan compenseren.

Ten slotte moeten we opmerken dat het functioneel en logisch ontwerp zoals hier beschreven, niet volledig zijn. Er zijn nadere eisen en wensen te stellen aan een systeem: performance, beveiliging, prijskaartje. Deze nadere vereisten moeten worden gespecificeerd naast het ontwerp zoals dat hier is geschetst. In onze optiek kan dat zonder de hier geschetste methode geweld aan te doen.

Ambler, S. (nog te verschijnen). The object primer (3rd edition), Agile Modeling Driven Development with UML 2.

Chen, P. (1976). The entity-relationship model - toward a unified view of data. In ACM Transactions on Database Systems, Vol. 1 (1): p. 9-36, 1976.

Dennis, A., B.H. Wixom & D. Tegarden (2002). Systems Analysis and Design, An Object-Oriented Approach with UML. John Wiley.

Eriksson, H-E. & M. Penker (2000). Business Modeling with UML, Business Patterns at Work. Prentice Hall.

Jeffries, R., A. Anderson & C. Hendrickson (2000). Extreme Programming Installed. Addison-Wesley.

Kristen, G. (1998). Kennis is Macht. Academic Service.

Larman, C. (2002). Applying UML and Patterns. Prentice Hall.

Morgan, G. (1997). Images of Organization. Sage Publications.

Vernadat, F.B. (1996). Enterprise modelling and integration: principles and applications. Chapmann & Hall.

Wetherbe, J.C. (1991). Executive Information Requirements: Getting it right. In MIS Quarterly. www.agilemodeling.com www.reengineeringllc.com www.xprogramming.com
Reviewer Stef Joosten 1. Een bonuskaart kan op verschillende manieren worden toegepast. In dit voorbeeld kiezen we voor de manier dat de bonuskaart gekoppeld is aan de klantnaam en de bonusprijs bij een aanbieding slechts geldt als de klant zijn bonuskaart kan laten zien.